BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. Tinjauan Pustaka

Dalam penelitian ini, penulis menggunakan beberapa sumber pustaka. Sumber pustaka yang dimaksud akan digunakan sebagai pedoman dan pembanding dalam penelitian yang akan penulis lakukan. Pustaka yang digunakan ditinjau dari segi objek penelitian, metode yang digunakan, serta hasil dan kesimpulan yang diperoleh dalam penelitian tersebut.

Penelitian tentang pengenalan pola huruf menggunakan metode *backpropagation* yang pernah dilakukan sebelumnya yaitu Sholahuddin (2002) dan Handoyo & Susanto (2011). Sholahuddin (2002) melakukan penelitian untuk mengenali pola huruf alfabet kapital (A-Z) dan berhasil membuat komputer mengenali huruf meskipun citra hurufnya diberi *noise*. Handoyo dan Susanto (2011) melakukan penelitian untuk mengenali pola huruf Jepang *katakana* dan *hiragana*. Mereka memberikan kesimpulan bahwa durasi pelatihan yang semakin lama akan semakin meningkatkan akurasi pengenalan. Metode ekstraksi fitur yang digunakan untuk kedua penelitian ini adalah binerisasi pola huruf.

Andrian (2012) telah melakukan penelitian untuk mengenali huruf hijaiyah menggunakan metode binerisasi pola huruf untuk mengekstraksi fitur dan *self-organizing map* untuk mengenali pola. Presentase keberhasilan pengenalan huruf yang diperoleh dari penggunaan metode ini cukup baik yaitu sebesar 77.14% dari jumlah 140 sampel huruf yang digunakan.

Penelitian untuk mengenali angka hijaiyah dilakukan oleh Hamdianah (2015) dengan menggabungkan metode *principal component analysis* dan *backpropagation.* Tingkat keberhasilan pengenalan dengan menggunakan metode ini sangat rendah hanya sebesar 28% dan disimpulkan bahwa metode *principal component analysis* tidak cocok digunakan untuk mengekstraksi fitur angka hijaiyah.

Syuhada (2015) telah melakukan penelitian mengenai pengenalan plat nomor kendaraan menggunakan metode histogram citra dan *backpropagation*. Metode ini menghasilkan tingkat rata-rata kesalahan (error) pelatihan 1.907% dan 1.963% serta tingkat akurasi pengujian sebesar 88% dan 60%. Pelatihan bertingkat (ganda) terhadap karakter yang memiliki kemiripan bentuk, akan menyebabkan waktu penerjemahan menjadi lebih lama dibandingkan dengan pelatihan tunggal tetapi akurasi pengenalan akan lebih akurat.

Perbandingan antara beberapa hasil penelitian bisa dilihat pada Tabel 2.1.

Tabel 2.1 Perbandingan Hasil Penelitian

Penulis dan Tahun	Objek	Metode Ektraksi Fitur	Metode Pengenalan Pola	Hasil
Asep Sholahuddin, MT (2002)	Huruf Alfabet	Binerisasi Pola Huruf	*Backpropagation*	*Backpropagation* mampu 100% mengenali huruf dari A sampai Z, bahkan ketika diberi noise sampai dengan rata-rata 0.05. Ketika diberi noise rata-rata 0.2, JST mulai membuat kesalahan.

Penulis dan Tahun	Objek	Metode Ektraksi Fitur	Metode Pengenalan Pola	Hasil
Erico Darmawan Handoyo & Lydia Wiguna Susanto (2011)	Huruf Jepang	Binerisasi Pola Huruf	*Backpropagation*	Presentase kemiripan input dengan hasil pengenalan rata-rata di atas 90% kecuali untuk huruf Fu yang hanya mencapai 37.8%. Durasi pelatihan jaringan yang semakin lama akan membuat nilai error semakin menipis, sehingga akan meningkatkan keakurasian
Weli Andrian (2012)	Huruf Hijaiyah	Binerisasi Pola Huruf	*Self-Organizing Map (Kohonen)*	Presentase keberhasilan pengenalan mencapai 77.14% dari 140 sampel huruf hijaiyah lepas
Muhammad Syuhada (2015)	Plat Nomor Kendaraan	Histogram Citra	*Backpropagation*	Tingkat rata-rata error pelatihan 1.907% dan 1.963% serta tingkat akurasi pengujian 88% dan 60%
Andi Hamdianah (2015)	Angka Hijaiyah	*Principal Component Analysis*	*Backpropagation*	Tingkat keberhasilan pengenalan sangat rendah, hanya sekitar 28%. Disimpulkan bahwa PCA kurang bisa mewakili ciri pola tulisan angka hijaiyah
Usulan Peneliti (2017)	Huruf Hijaiyah Tulisan Tangan	Binerisasi Pola Huruf	*Backpropagation*	Nilai akurasi yang diperoleh sebesar 51.33% dari 150 sampel data uji. Jumlah neuron pada hidden layer dan target error mempengaruhi akurasi sistem.

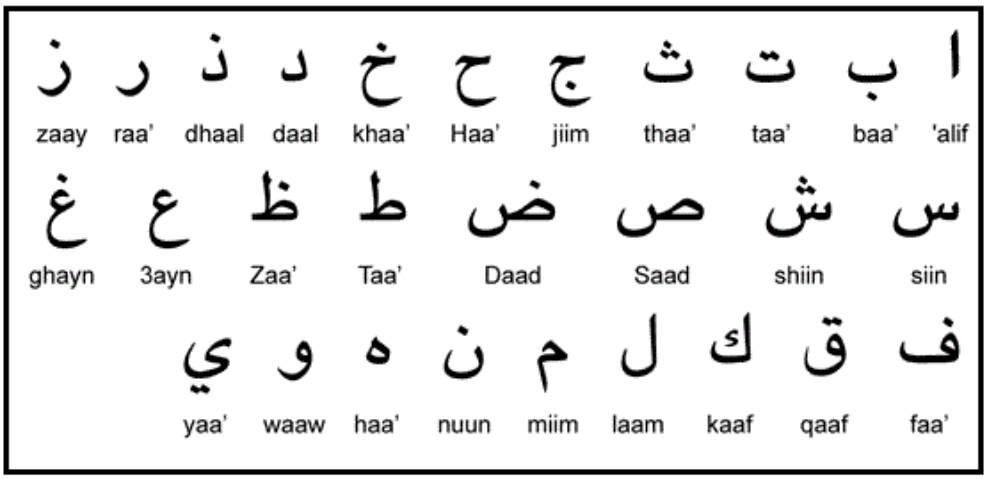
2.2. Dasar Teori

2.2.1. Huruf Hijaiyah

Dalam Kamus Besar Bahasa Indonesia kata hijaiyah berarti "Sistem aksara Arab; Abjad Arab". Kata *huruf* berasal dari bahasa arab *harf* atau *huruuf* (حرف او حروف). Huruf arab disebut juga huruf *hija'iyah* (هجائية) . Kata *hija'iyah* berasal dari kata kerja *hajjaa* (هجى) yang artinya *mengeja, menghitung huruf, membaca huruf demi huruf* (Muhammad, 1990). Huruf *hija'iyah* disebut pula *huruuf tahjiyyah* (حروف تهجية) .

Huruf *hijaiyah* disebut juga *alfabet arab*. Kata *alfabet* itu sendiri berasal dari bahasa arab *alif, ba', ta'* (Husain, 1988). Kata *abjad* juga berasal dari bahasa arab *a-ba-ja-dun*; *alif, ba', ta', jim,* dan *dal* (أبجد) . Namun ada pula yang menolak pendapat ini dengan alasan, huruf hijaiyah mempunyai aturan urutan yang berbeda dengan terminologi abjad. Huruf hijaiyah dimulai dari *alif* dan berakhir pada huruf *ya'* secara terpisah-pisah. Sedang terminologi abjad urutannya disusun dalam bentuk kalimat (أبجد هوز حطى كلمن سعفص قرشت), di samping itu terminologi abjad lebih bersifat terbatas pada bahasa *samiyah* yang lokal (*lughah samiyah al-umm*) (Thohir dan Nabawi, 1987).

Huruf hijaiyah berjumlah 28 huruf tunggal atau 30 jika memasukkan huruf rangkap *lam-alif* (لا) dan *hamzah* (ء) sebagai huruf yang berdiri sendiri. Orang yang pertama kali menyusun huruf *hijaiyah* secara berurutan mulai dari *alif* sampai *ya'* adalah Nashr Bin 'Ashim Al-Laitsi (ناصر بن عاصم الليثي). Cara menulis huruf Arab berbeda dengan huruf Latin, yaitu ditulis dari kanan ke kiri (Hitti, 2005). Bentuk-bentuk huruf hijaiyah dasar bisa dilihat pada Gambar 2.1.



Sumber : www.myeasyarabic.com

Gambar 2.1 Huruf Hijaiyah Dasar

2.2.2. Pengolahan Citra dan Ekstraksi Fitur

Pengolahan citra adalah istilah umum untuk berbagai teknik yang keberadaannya untuk memanipulasi dan memodifikasi citra dengan berbagai cara. Foto adalah contoh gambar berdimensi dua yang dapat diolah dengan mudah. Setiap foto dalam bentuk citra digital (misalkan berasal dari kamera digital) dapat diolah melalui perangkat lunak tertentu (Kadir, 2013). Sebagai contoh, apabila hasil bidikan kamera terlihat agak gelap, citra dapat diolah menjadi lebih terang. Dimungkinkan pula untuk memisahkan foto orang dari latar belakangnya. Gambaran tersebut menunjukkan contoh hal sederhana yang dapat dilakukan melalui pengolahan citra digital.

Fitur merupakan karakteristik unik dari suatu objek. Ekstraksi fitur adalah proses untuk mendapatkan ciri-ciri pembeda yang membedakan suatu objek dari objek yang lain (Putra, 2010). Ekstraksi fitur citra terbagi menjadi 3 macam, yaitu ekstraksi fitur bentuk, tekstur, dan warna. Bentuk dari suatu objek adalah karakter konfigurasi permukaan yang diwakili oleh garis dan kontur, inilah yang diambil pada ekstraksi fitur bentuk. Pada ekstraksi fitur tekstur, fitur pembeda adalah tekstur yang merupakan karakteristik penentu pada citra. Pada ekstraksi fitur warna, ciri pembeda adalah warna. Biasanya ekstraksi fitur ini digunakan pada citra berwarna yang memiliki komposisi warna RGB (*red*, *green*, *blue*). Namun juga bisa dilakukan ekstraksi fitur pada citra keabuan (*grayscale*) maupun citra biner (*black-white*).

2.2.3. Citra Biner

Citra biner (*binary image)* adalah citra yang hanya mempunyai dua nilai derajat keabuan: hitam dan putih. Citra biner banyak digunakan misalnya pada citra logo instansi (yang hanya memiliki warna hitam dan putih), citra kode batang (*barcode*), citra hasil pemindaian (*scanner*), dan lain-lain (Putra, 2010). Pada citra biner, piksel-piksel objek bernilai 1 dan piksel-piksel latar belakang (*background*) bernilai 0. Pada waktu menampilkan citra, 0 adalah hitam dan 1 adalah putih. Jadi pada citra biner, latar belakang berwarna hitam sedangkan objek berwarna putih.

Penggunaan citra biner memiliki kelebihan yaitu kebutuhan memorinya kecil karena nilai derajat keabuan hanya mempunyai representasi 1 bit. Hal ini juga menyebabkan waktu pemrosesan citra lebih cepat jika dibandingkan dengan citra berwarna dan citra *grayscale*.

Pengkonversian citra berwarna dan citra *grayscale* menjadi citra biner dilakukan untuk alasan-alasan berikut:

- Untuk menampilkan citra pada piranti keluaran yang hanya mempunyai resolusi intensitas satu bit.
- Untuk mengindentifikasi keberadaan objek.
- Untuk lebih memfokuskan pada analisis bentuk morfologi.

2.2.4. Jaringan Syaraf Tiruan (JST)

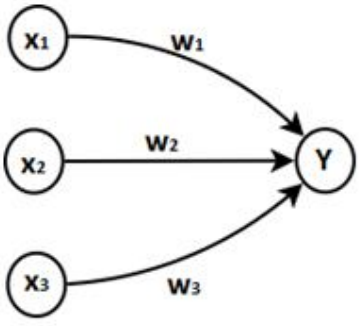
Jaringan Syaraf Tiruan (JST) adalah sistem pemroses informasi yang memiliki karakteristik mirip dengan jaringan syaraf biologi (Siang, 2005). JST dibentuk sebagai generalisasi model matematika dari jaringan syaraf biologi, dengan asumsi bahwa:

- Pemrosesan infromasi terjadi pada banyak elemen sederhana (neuron)
- Sinyal dikirimkan diantara neuron-neuron melalui penghubung-penghubung
- Penghubung antar neuron memiliki bobot yang akan memperkuat atau memperlemah sinyal
- Untuk menentukan output, setiap neuron menggunakan fungsi aktivasi yang dikenakan pada penjumlahan input yang diterima.

Jaringan syaraf tiruan dapat digunakan untuk mengenali pola (misal huruf, angka, suara, atau tanda tangan) yang sudah sedikit berubah. Hal ini mirip dengan otak manusia yang masih mampu mengenali orang yang sudah beberapa waktu tidak dijumpainya (mungkin wajah/bentuk tubuh nya sudah sedikit berubah).

Jaringan syaraf tiruan ditentukan oleh 3 hal:

- Pola hubungan antar neuron
- Metode untuk menentukan bobot penghubung (disebut metode *training* / *learning* / algoritma)
- Fungsi aktivasi



Sumber : (Siang, 2005)

Gambar 2.2 Arsitektur Jaringan Syaraf Tiruan Sederhana

Perhatikan Gambar 2.2, Y akan menerima input dari neuron x_1, x_2, dan x_3 dengan bobot hubungan masing-masing adalah w_1, w_2, dan w_3. Ketiga impuls neuron yang ada lalu dijumlahkan. Besarnya impuls yang diterima oleh Y mengikuti fungsi aktivasi tertentu. Apabila nilai fungsi aktivasi cukup kuat, maka sinyal akan diteruskan.

2.2.5. Backpropagation

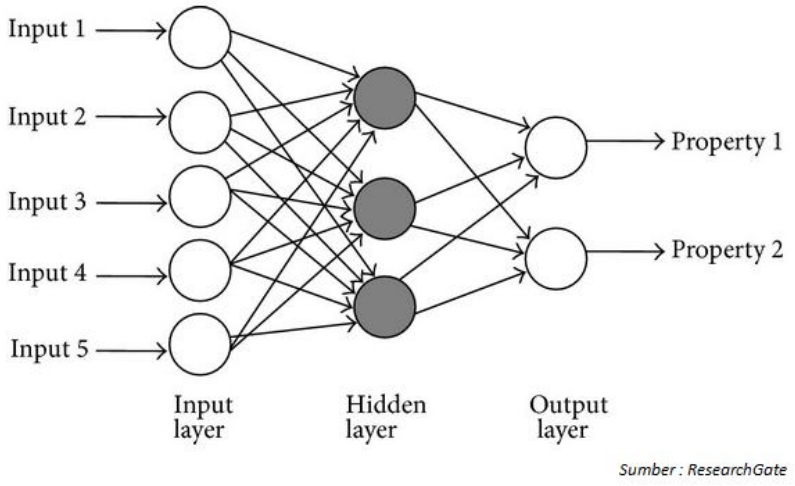
JST dengan *layer* (lapisan) tunggal memiliki keterbatasan dalam pengenalan pola. Kelemahan ini bisa ditanggulangi dengan menambahkan satu atau beberapa layer tersembunyi (*hidden layer*) diantara layer *input* (masukan) dan layer *output* (keluaran). Seperti halnya model JST yang lain, *backpropagation* melatih jaringan untuk mendapatkan keseimbangan antara kemampuan jaringan

untuk mengenali pola masukan yang serupa (tapi tidak sama) dengan pola yang dipakai selama pelatihan (Siang, 2005).

Backpropagation merupakan algoritma pembelajaran yang terawasi dan biasanya digunakan oleh model jaringan syaraf tiruan *perceptron* dengan banyak lapisan *(multi-layered perceptron)* untuk mengubah bobot–bobot yang terhubung dengan neuron–neuron yang ada pada lapisan tersembunyinya (Kusumadewi, 2004). Ciri dari metode backpropagation adalah meminimalkan error pada output yang dihasilkan oleh jaringan. Arsitektur jaringan syaraf tiruan dengan metode backpropagation terdiri dari:

- Satu layer input
- Satu atau lebih layer tersembunyi (*hidden layer*)
- Satu layer output

Jumlah neuron pada lapisan masukan sama dengan jumlah ciri atau fitur pada pola yang akan dikenali, sedangkan jumlah neuron pada lapisan keluaran sama dengan jumlah kelas pola. Arsitektur JST *backpropagation* bisa dilihat pada Gambar 2.3.



Gambar 2.3 Arsitektur JST Backpropagation

Pelatihan backpropagation meliputi 3 fase. Fase pertama adalah fase maju. Pola masukan dihitung maju mulai dari layer input hingga layer output menggunakan fungsi aktivasi. Fungsi aktivasi yang paling umum digunakan untuk backpropagation adalah fungsi sigmoid biner. Fase kedua adalah fase mundur. Selisih antara output jaringan dengan target yang diinginkan merupakan kesalahan yang terjadi. Kesalahan tersebut dipropagasikan mundur dimulai dari garis yang berhubungan langsung dengan unit-unit di layer output. Fase ketiga adalah modifikasi bobot yang menurunkan kesalahan yang terjadi.

Algoritma *Backpropagation*:

1. Inisialisasi bobot (ambil bobot awal dengan nilai random yang cukup kecil).
2. Tetapkan maksimum Epoch, Target Error, dan Learning Rate (α).
3. Inisialisasi: Epoch = 0, Error = 1.
4. Kerjakan langkah-langkah berikut selama (Epoch < Maksimum Epoch) dan (Error > Target Error):
   1) Epoch = Epoch + 1
   2) Untuk tiap-tiap pasangan elemen yang akan dilakukan pembelajaran, kerjakan:

Feedforward:

(a) Tiap-tiap unit input (X_i, i=1,2,3,…,n) menerima sinyal x_i dan meneruskan sinyal tersebut ke semua unit pada lapisan yang ada diatasnya (lapisan tersembunyi).

(b) Tiap-tiap unit pada suatu lapisan tersembunyi (Z_j, j=1,2,3,…,p) menjumlahkan sinyal-sinyal input terbobot, seperti pada rumus 2.1:

$$z\_in_j = b1_j + \sum_{i=1}^{n} x_i v_{ij} \quad \text{..... 2.1}$$

Gunakan fungsi aktivasi untuk menghitung sinyal outputnya, bisa dilihat pada rumus 2.2:

$$z_j = f(z\_in_j) \quad \text{..... 2.2}$$

Dan kirimkan sinyal tersebut ke semua unit di lapisan atasnya (unit-unit output).

Langkah (b) ini dilakukan sebanyak jumlah lapisan tersembunyi.

(c) Tiap-tiap unit output (Y_k, k=1,2,3,…,m) menjumlahkan sinyal-sinyal input terbobot, lihat pada rumus 2.3:

$$y\_in_k = b2_k + \sum_{i=1}^{p} z_i w_{jk} \quad \text{..... 2.3}$$

Gunakan fungsi aktivasi untuk menghitung sinyal outputnya, sesuai dengan rumus 2.4:

$$y_k = f(y\_in_k) \quad \text{..... 2.4}$$

Backforward:

(d) Tiap-tiap unit output (Y_k, k=1,2,3,…,m) menerima target pola yang berhubungan dengan pola input pembelajaran, hitung informasi errornya menggunakan turunan dari fungsi aktivasi, seperti pada rumus 2.5, 2.6, dan 2.7:

$$\delta 2_k = (t_k - y_k) f'(y\_in_k) \quad \text{..... 2.5}$$

$$\varphi 2_{jk} = \delta_k z_j \quad \text{..... 2.6}$$

$$\boldsymbol{\beta 2_k = \delta_k} \quad \text{..... 2.7}$$

Kemudian hitung koreksi bobot (yang nantinya akan digunakan untuk memperbaiki nilai w_{ij}) dengan rumus 2.8:

$$\boldsymbol{\Delta w_{jk} = \alpha \varphi 2_{jk}} \quad \text{..... 2.8}$$

Hitung juga koreksi bias (yang nantinya akan digunakan untuk memperbaiki nilai $b2_k$) dengan rumus 2.9:

$$\boldsymbol{\Delta b2_k = \alpha \beta 2_k} \quad \text{..... 2.9}$$

Langkah (d) ini juga digunakan sebanyak jumlah lapisan tersembunyi, yaitu menghitung informasi error dari suatu lapisan tersembunyi ke lapisan tersembunyi sebelumnya.

(e) Tiap-tiap unit tersembunyi (Z_j, j=1,2,3,…,p) menjumlahkan delta inputnya (dari unit-unit yang berada pada lapisan atasnya) sererti pada rumus 2.10:

$$\boldsymbol{\delta\_in_j = \sum_{k=1}^{m} \delta 2_k w_{jk}} \quad \text{..... 2.10}$$

Kalikan nilai ini dengan turunan dari fungsi aktivasi untuk menghitung informasi error, bisa dilihat pada rumus 2.11, 2.12, dan 2.13:

$$\boldsymbol{\delta 1_j = \delta\_in_j \, f'(z\_in_j)} \quad \text{..... 2.11}$$

$$\boldsymbol{\varphi 1_{ij} = \delta 1_j x_j} \quad \text{..... 2.12}$$

$$\boldsymbol{\beta 1_j = \delta 1_j} \quad \text{..... 2.13}$$

Kemudian hitung koreksi bobot (yang nantinya digunakan untuk memperbaiki nilai v_{ij}) seperti terlihat pada rumus 2.14:

$$\boldsymbol{\Delta v_{ij} = \alpha \varphi 1_{ij}} \quad \text{..... 2.14}$$

Hitung juga koreksi bias (yang nantinya akan digunakan untuk memperbaiki nilai $b1_j$) seperi terlihat pada rumus 2.15:

$$\Delta \boldsymbol{b1_j} = \boldsymbol{\alpha\beta 1_j} \quad \text{..... 2.15}$$

(f) Tiap-tiap unit output (Y_k, k=1,2,3,…,m) memperbaiki bias dan bobotnya (j=0,1,2,…,p) dengan rumus 2.16 dan 2.17:

$$\mathbf{w_{jk}(baru) = w_{jk}(lama) + \Delta w_{jk}} \quad \text{..... 2.16}$$

$$\mathbf{b2_k(baru) = b2_k(lama) + \Delta b2_k} \quad \text{..... 2.17}$$

tiap-tiap unit tersembunyi (Z_j, j=1,2,3,…,p) memperbaiki bias dan bobotnya (i=0,1,2,…,n) dengan rumus 2.18 dan 2.19:

$$\mathbf{v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij}} \quad \text{..... 2.18}$$

$$\mathbf{b1_j(baru) = b1_j(lama) + \Delta b1_j} \quad \text{..... 2.19}$$

(g) Hitung nilai Error menggunakan rumus *mean squared error (MSE)* seperti pada rumus 2.20:

$$\boldsymbol{E = \frac{1}{n} \sum_{i=1}^{n} (t_k - y_k)^2} \quad \text{..... 2.20}$$

Mean Squared Error (MSE) adalah fungsi yang digunakan untuk mengukur kesalahan (error) dalam proses pelatihan jaringan. Nilai error dihitung dari selisih antara nilai output yang didapat dengan nilai target yang ditetapkan, kemudian hasilnya di kuadratkan. Nilai error ini akan menjadi acuan untuk mengukur kinerja pelatihan jaringan, semakin kecil nilai error artinya semakin baik hasil pelatihannya. Pelatihan jaringan akan berhenti hanya ketika target error sudah tercapai atau ketika iterasi sudah mencapai batas.

Fungsi aktivasi merupakan operasi matematik yang dikenakan pada sinyal output y. Fungsi ini digunakan untuk mengaktifkan atau menonaktifkan neuron.

Perilaku dari JST ditentukan oleh bobot dan input-input fungsi aktivasi yang ditetapkan. Fungsi aktivasi yang sering digunakan untuk melatih jaringan syaraf tiruan menggunakan metode *backpropagation* adalah fungsi sigmoid biner. Fungsi sigmoid biner memiliki nilai pada range 0 sampai 1. Oleh karena itu, fungsi ini sering digunakan untuk jaringan yang membutuhkan nilai output yang terletak pada interval 0 sampai 1. Namun, fungsi ini bisa juga digunakan oleh jaringan syaraf yang nilai outputnya 0 atau 1 (Kusumadewi, 2004). Fungsi sigmoid biner seperti terlihat pada rumus 2.21:

$$y = f(x) = \frac{1}{1+e^{-\sigma x}} \quad \text{..... 2.21}$$

Dengan turunannya pada rumus 2.22:

$$f'(x) = \sigma f(x)[1 - f(x)] \quad \text{..... 2.22}$$

2.2.6. Akurasi

Sebuah sistem yang melakukan identifikasi atau klasifikasi diharapkan mampu melakukan klasifikasi semua data set dengan benar. Namun tidak dapat dipungkiri bahwa kinerja suatu sistem yang melakukan klasifikasi tidak akan selalu bisa 100% benar. Oleh karena itu, sistem harus diukur kinerjanya. Umumnya cara mengukur kinerja klasifikasi menggunakan *confusion matrix* (Prasetyo, 2014).

Pengukuran terhadap kinerja suatu sistem klasifikasi merupakan hal yang penting. Kinerja sistem klasifikasi menggambarkan seberapa baik sistem dalam mengklasifikasikan data. Confusion matrix merupakan salah satu metode yang dapat digunakan untuk mengukur kinerja suatu metode klasifikasi. Pada dasarnya

confusion matrix mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya.

Pada pengukuran kinerja menggunakan confusion matrix, terdapat 4 (empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah True Positive (TP), True Negative (TN), False Positive (FP) dan False Negative (FN). Nilai True Negative (TN) merupakan jumlah data negatif yang terdeteksi dengan benar, sedangkan False Positive (FP) merupakan data negatif namun terdeteksi sebagai data positif. Sementara itu, True Positive (TP) merupakan data positif yang terdeteksi benar. False Negative (FN) merupakan kebalikan dari True Positive, sehingga data positif, namun terdeteksi sebagai data negatif. Pada jenis klasifikasi biner yang hanya memiliki 2 keluaran kelas, confusion matrix dapat disajikan seperti pada Tabel 2.2.

Tabel 2.2 Confusion Matrix Klasifikasi Biner

Kelas	Terklasifikasi Positif	Terklasifikasi Negatif
Positif	TP (True Positive)	FN (False Negative)
Negatif	FP (False Positive)	TN (True Negative)

Berdasarkan nilai True Negative (TN), False Positive (FP), False Negative (FN), dan True Positive (TP) dapat diperoleh nilai akurasi. Nilai akurasi menggambarkan seberapa akurat sistem dapat mengklasifikasikan data secara benar. Dengan kata lain, nilai akurasi merupakan perbandingan antara data yang terklasifikasi benar dengan keseluruhan data. Nilai akurasi dapat diperoleh dengan

rumus 2.23 di bawah ini:

$$Akurasi = \frac{jumlah\ data\ yang\ diprediksi\ secara\ benar}{jumlah\ pengujian\ yang\ dilakukan}\ x\ 100 \quad \text{..... 2.23}$$

2.2.7. MATLAB

Matrix Laboratory atau yang biasa disebut dengan MATLAB adalah sebuah program untuk menganalisis dan mengkomputasi data numerik, dan merupakan suatu bahasa pemrograman matematika lanjutan, yang dibentuk dengan dasar pemikiran yang menggunakan sifat dan bentuk matriks.

Matlab dikembangkan oleh The Mathwork Inc. yang hadir dengan fungsi dan karakteristik yang berbeda dengan bahasa pemrograman lain yang sudah ada lebih dahulu seperti Delphi, Basic maupun C++. Matlab merupakan bahasa canggih untuk komputasi teknik. Matlab merupakan integrasi dari komputansi, visualisasi dan pemograman dalam suatu lingkungan yang mudah digunakan, karena permasalahan dan pemecahannya dinyatakan dalam notasi matematika biasa (Pujiriyanto:2004). Kegunaan Matlab secara umum adalah untuk :

- Matematika dan Komputasi
- Pengembangan dan Algoritma
- Pemodelan, simulasi dan pembuatan prototype
- Analisis Data,eksplorasi dan visualisasi
- Pembuatan apilikasi termasuk pembuatan graphical *user* interface

Matlab adalah sistem interaktif dengan elemen dasar array yang merupakan basis datanya. Array tersebut tidak perlu dinyatakan khusus seperti di

bahasa pemograman yang ada sekarang. Hal ini memungkinkan kita untuk memecahkan banyak masalah perhitungan teknik, khususnya yang melibatkan matriks dan vektor dengan waktu yang lebih singkat dari waktu yang dibutuhkan untuk menulis program dalam bahasa C atau Fortran. Untuk memahami matlab, kita harus sudah paham terlebih dahulu mengenai matematika terutama operasi vektor dan matriks, karena operasi matriks merupakan inti utama dari matlab (Pujiriyanto, 2004).

Pada intinya matlab merupakan sekumpulan fungsi-fungsi yang dapat dipanggil dan dieksekusi. Fungsi-fungsi tersebut dibagi-bagi berdasarkan kegunaannya yang dikelompokan didalam toolbox yang ada pada matlab. Untuk mengetahui lebih jauh mengenai toolbox yang ada di matlab dan fungsinya kita dapat mencari informasinya di website www.mathworks.com.